

***Security at the  
Application Layer:  
PGP and S/MIME***

# Objectives

- ☐ To explain the general structure of an e-mail application program
- ☐ To discuss how PGP can provide security services for e-mail
- ☐ To discuss how S/MIME can provide security services for e-mail
- ☐ To define trust mechanism in both PGP and S/MIME
- ☐ To show the structure of messages exchanged in PGP and S/MIME

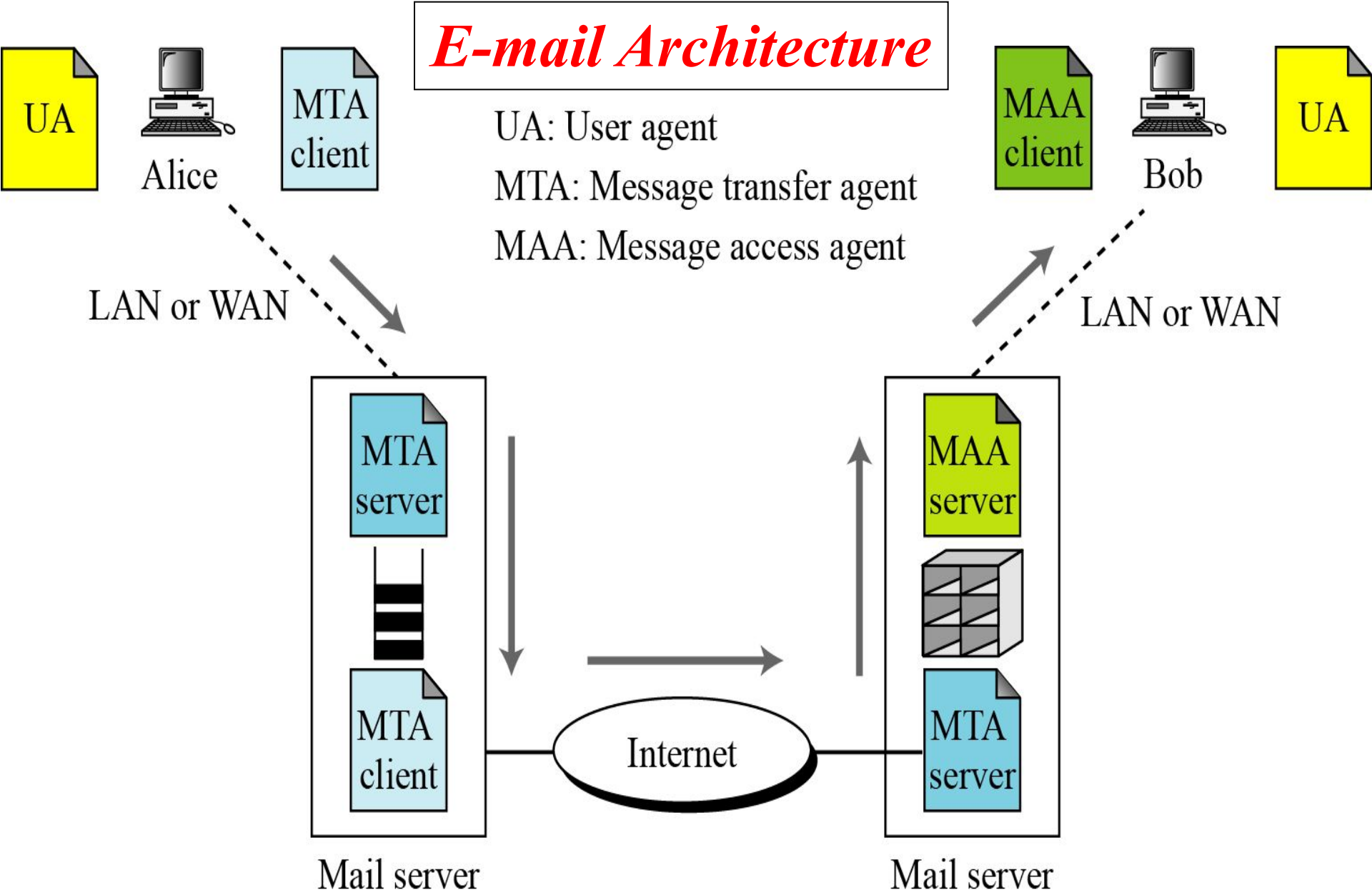
# E-MAIL

**Let us first discuss the electronic mail (e-mail) system in general.**

**Topics discussed in this section:**

**16.1.1 E-mail Architecture**

**16.1.2 E-mail Security**





# *E-mail Security*

## *Cryptographic Algorithms*

### *Note*

**In e-mail security, the sender of the message needs to include the name or identifiers of the algorithms used in the message.**

## *Certificates*

*It is obvious that some public-key algorithms must be used for e-mail security.*

**In e-mail security, the encryption/decryption is done using a symmetric-key algorithm, but the secret key to decrypt the message is encrypted with the public key of the receiver and is sent with the message.**

# PGP

**Pretty Good Privacy (PGP) can be used to create a secure e-mail message or to store a file securely for future retrieval.**

**Topics discussed in this section:**

**16.2.1 Scenarios**

**16.2.2 Key Rings**

**16.2.3 PGP Certificates**

**16.2.4 Key Revocation**

**16.2.5 Extracting Information from Rings**

**16.2.6 PGP Packets**

**16.2.7 PGP Messages**



# *A plaintext message*

Alice



Bob

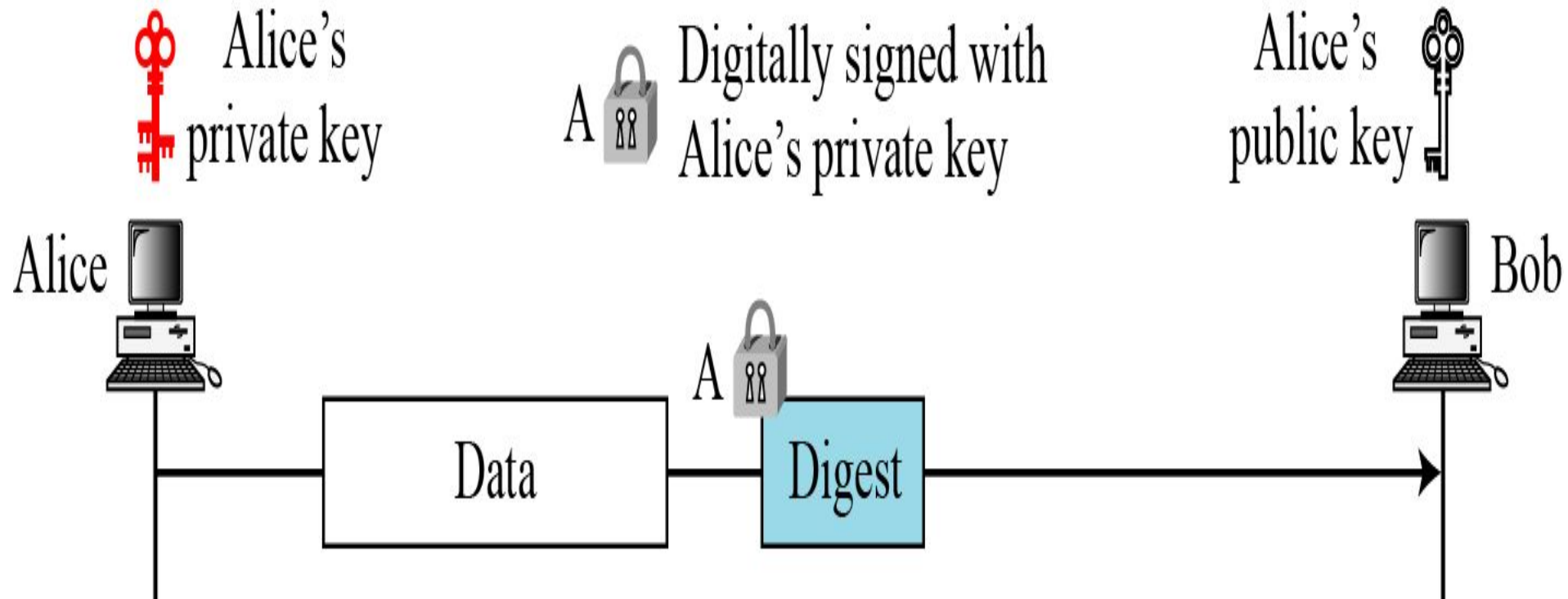


Data

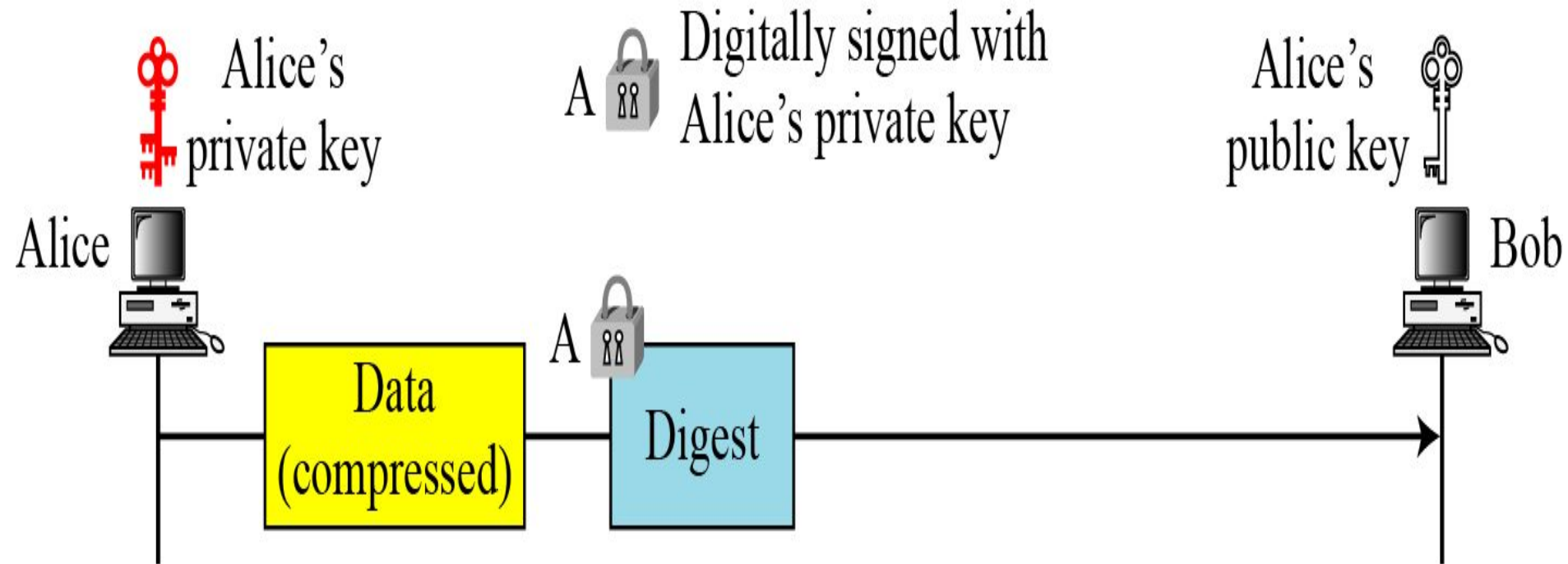


## Message Integrity

# An authenticated message

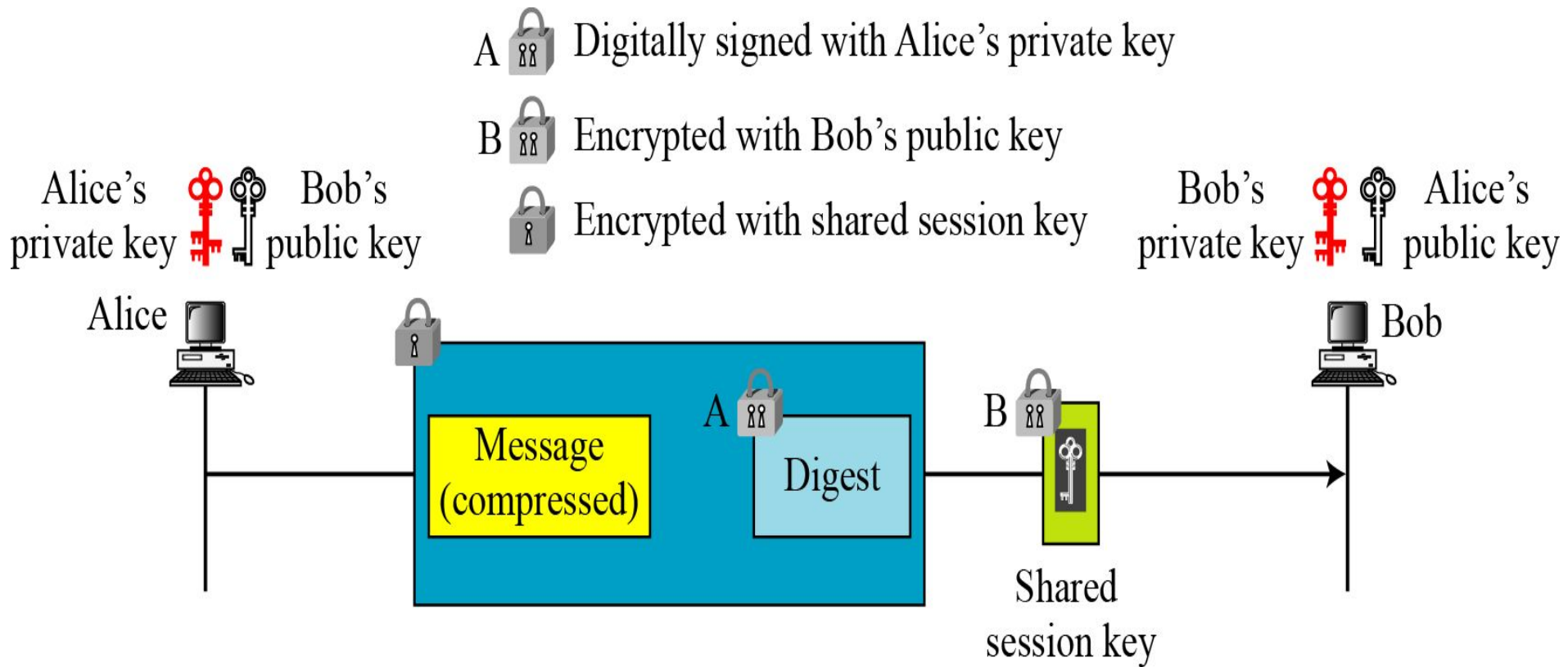


# A compressed message



# Confidentiality with One-Time Session Key

## *A confidential message*





## *16.2.1 Continued*

---

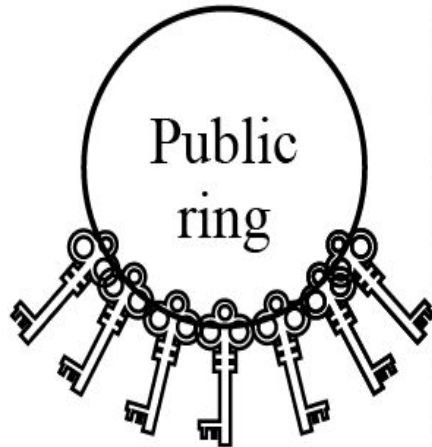
- **Code Conversion**
- **Another service provided by PGP is code conversion. PGP uses Radix-64 conversion.**
- **Segmentation**
- **PGP allows segmentation of the message.**

Index	Binary	Char	Index	Binary	Char	Index	Binary	Char	Index	Binary	Char
0	000000	A	16	010000	Q	32	100000	g	48	110000	w
1	000001	B	17	010001	R	33	100001	h	49	110001	x
2	000010	C	18	010010	S	34	100010	i	50	110010	y
3	000011	D	19	010011	T	35	100011	j	51	110011	z
4	000100	E	20	010100	U	36	100100	k	52	110100	θ
5	000101	F	21	010101	V	37	100101	l	53	110101	1
6	000110	G	22	010110	W	38	100110	m	54	110110	2
7	000111	H	23	010111	X	39	100111	n	55	110111	3
8	001000	I	24	011000	Y	40	101000	o	56	111000	4
9	001001	J	25	011001	Z	41	101001	p	57	111001	5
10	001010	K	26	011010	a	42	101010	q	58	111010	6
11	001011	L	27	011011	b	43	101011	r	59	111011	7
12	001100	M	28	011100	c	44	101100	s	60	111100	8
13	001101	N	29	011101	d	45	101101	t	61	111101	9
14	001110	O	30	011110	e	46	101110	u	62	111110	+
15	001111	P	31	011111	f	47	101111	v	63	111111	/
Padding		=									

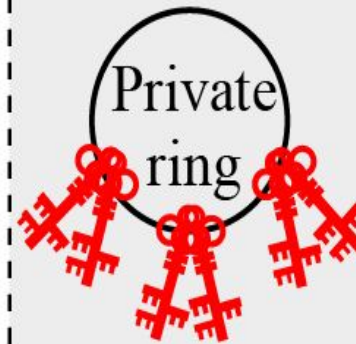
R  
A  
D  
I  
V

# Key rings in PGP

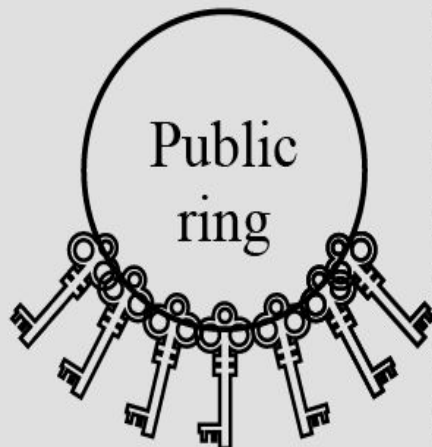
Alice's rings



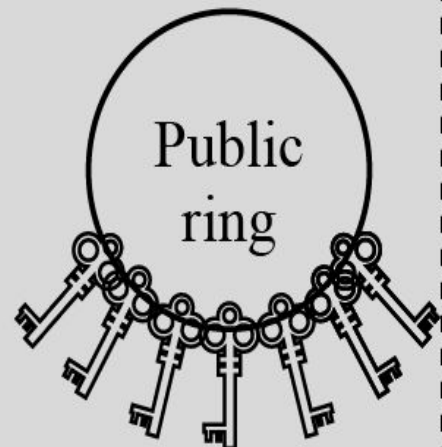
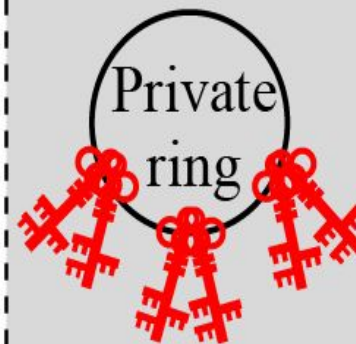
Bob's rings



Ted's rings



John's rings





## 16.2.2 Continued

### *PGP Algorithms*

**Table 16.1** *Public-key algorithms*

<i>ID</i>	<i>Description</i>
1	RSA (encryption or signing)
2	RSA (for encryption only)
3	RSA (for signing only)
16	ElGamal (encryption only)
17	DSS
18	Reserved for elliptic curve
19	Reserved for ECDSA
20	ElGamal (for encryption or signing)
21	Reserved for Diffie-Hellman
100–110	Private algorithms



## 16.2.2 Continued

**Table 16.2** *Symmetric-key algorithms*

<i>ID</i>	<i>Description</i>
0	No Encryption
1	IDEA
2	Triple DES
3	CAST-128
4	Blowfish
5	SAFER-SK128
6	Reserved for DES/SK
7	Reserved for AES-128
8	Reserved for AES-192
9	Reserved for AES-256
100–110	Private algorithms



## 16.2.2 Continued

**Table 16.3** *Hash Algorithms*

<i>ID</i>	<i>Description</i>
1	MD5
2	SHA-1
3	RIPE-MD/160
4	Reserved for double-width SHA
5	MD2
6	TIGER/192
7	Reserved for HAVAL
100–110	Private algorithms

**Table 16.4** *Compression methods*

<i>ID</i>	<i>Description</i>
0	Uncompressed
1	ZIP
2	ZLIP
100–110	Private methods

## **X.509 Certificates**

**Protocols that use X.509 certificates depend on the hierarchical structure of the trust.**



### *Note*

**In X.509, there is a single path from the fully trusted authority to any certificate.**

### PGP Certificates

In PGP, there is no need for CAs; anyone in the ring can sign a certificate for anyone else in the ring.



#### *Note*

In PGP, there can be multiple paths from fully or partially trusted authorities to any subject.

### Trusts and Legitimacy

The entire operation of PGP is based on **introducer trust**, the **certificate trust**, and the **legitimacy of the public keys**.

## 16.2.3 Continued

**Figure 16.7** *Format of private key ring table*



User ID	Key ID	Public key	Encrypted private key	Timestamp
⋮	⋮	⋮	⋮	⋮

## 16.2.3 Continued

### Example 16.1

Let us show a private key ring table for Alice. We assume that Alice has only two user IDs, **alice@some.com** and **alice@anet.net**. We also assume that Alice has two sets of private/public keys, one for each user ID.

**Table 16.5**     *Private key ring table for Example 1*

<i>User ID</i>	<i>Key ID</i>	<i>Public Key</i>	<i>Encrypted Private Key</i>	<i>Timestamp</i>
alice@anet.net	AB13...45	AB13...45...59	<b>32452398...23</b>	031505-16:23
alice@some.com	FA23...12	FA23...12...22	<b>564A4923...23</b>	031504-08:11

## 16.2.3 Continued

**Figure 16.8** *Format of a public key ring table*



Public ring

User ID	Key ID	Public key	Producer trust	Certificate(s)	Certificate trust(s)	Key Legitimacy	Timestamp
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

## 16.2.3 Continued

### Example 16.2

**A series of steps will show how a public key ring table is formed for Alice.**

**Table 16.6** *Example 2, starting table*

<i>User ID</i>	<i>Key ID</i>	<i>Public key</i>	<i>Prod. trust</i>	<i>Certificate</i>	<i>Cert. trust</i>	<i>Key legit.</i>	<i>Time-stamp</i>
Alice...	AB...	AB.....	F			F	.....

**Table 16.7** *Example 2, after Bob is added to the table*

<i>User ID</i>	<i>Key ID</i>	<i>Public key</i>	<i>Prod. trust</i>	<i>Certificate</i>	<i>Cert. trust</i>	<i>Key legit.</i>	<i>Time-stamp</i>
Alice...	AB...	AB.....	F			F	.....
Bob...	12...	12.....	F			F	.....



## 16.2.3 Continued

### Example 16.2 Continued

**Table 16.8** *Example 2, after Ted is added to the table*

<i>User ID</i>	<i>Key ID</i>	<i>Public key</i>	<i>Prod. trust</i>	<i>Certificate</i>	<i>Cert. trust</i>	<i>Key legit.</i>	<i>Time-stamp</i>
Alice...	AB...	AB.....	F			F	.....
Bob...	12...	12.....	F			F	.....
Ted...	48...	48.....	F	Bob's	F	F	.....

**Table 16.9** *Example 2, after Anne is added to the table*

<i>User ID</i>	<i>Key ID</i>	<i>Public key</i>	<i>Prod. trust</i>	<i>Certificate</i>	<i>Cert. trust</i>	<i>Key legit.</i>	<i>Time-stamp</i>
Alice...	AB...	AB.....	F			F	.....
Bob...	12...	12.....	F			F	.....
Ted...	48...	48.....	F	Bob's	F	F	.....
Anne...	71...	71.....	P	Bob's	F	F	.....

## 16.2.3 Continued

### Example 16.2 Continued

**Table 16.10** *Example 2, after John is added to the table*

<i>User ID</i>	<i>Key ID</i>	<i>Public key</i>	<i>Prod. Trust</i>	<i>Certificate</i>	<i>Cert. trust</i>	<i>Key legit.</i>	<i>Time-stamp</i>
Alice...	AB...	AB.....	F			F	.....
Bob...	12...	12.....	F			F	.....
Ted...	48...	48.....	F	Bob's	F	F	.....
Anne...	71...	71.....	P	Bob's	F	F	.....
John...	31...	31.....	N	Anne's	P	P	.....

## 16.2.3 Continued

### Example 16.2 Continued

**Table 16.11** *Example 2, after one more certificate received for John*

<i>User ID</i>	<i>Key ID</i>	<i>Public key</i>	<i>Prod. trust</i>	<i>Certificate</i>	<i>Cert. trust</i>	<i>Key legit.</i>	<i>Time-stamp</i>
Alice...	AB...	AB.....	F			F	.....
Bob...	12...	12.....	F			F	.....
Ted...	48...	48.....	F	Bob's	F	F	.....
Anne...	71...	71.....	P	Bob's	F	F	.....
John...	31...	31.....	N	Anne's Ted's	P F	F	.....

## 16-3 S/MIME

Another security service designed for electronic mail is Secure/Multipurpose Internet Mail Extension (S/MIME). The protocol is an enhancement of the Multipurpose Internet Mail Extension (MIME) protocol.

*Topics discussed in this section:*

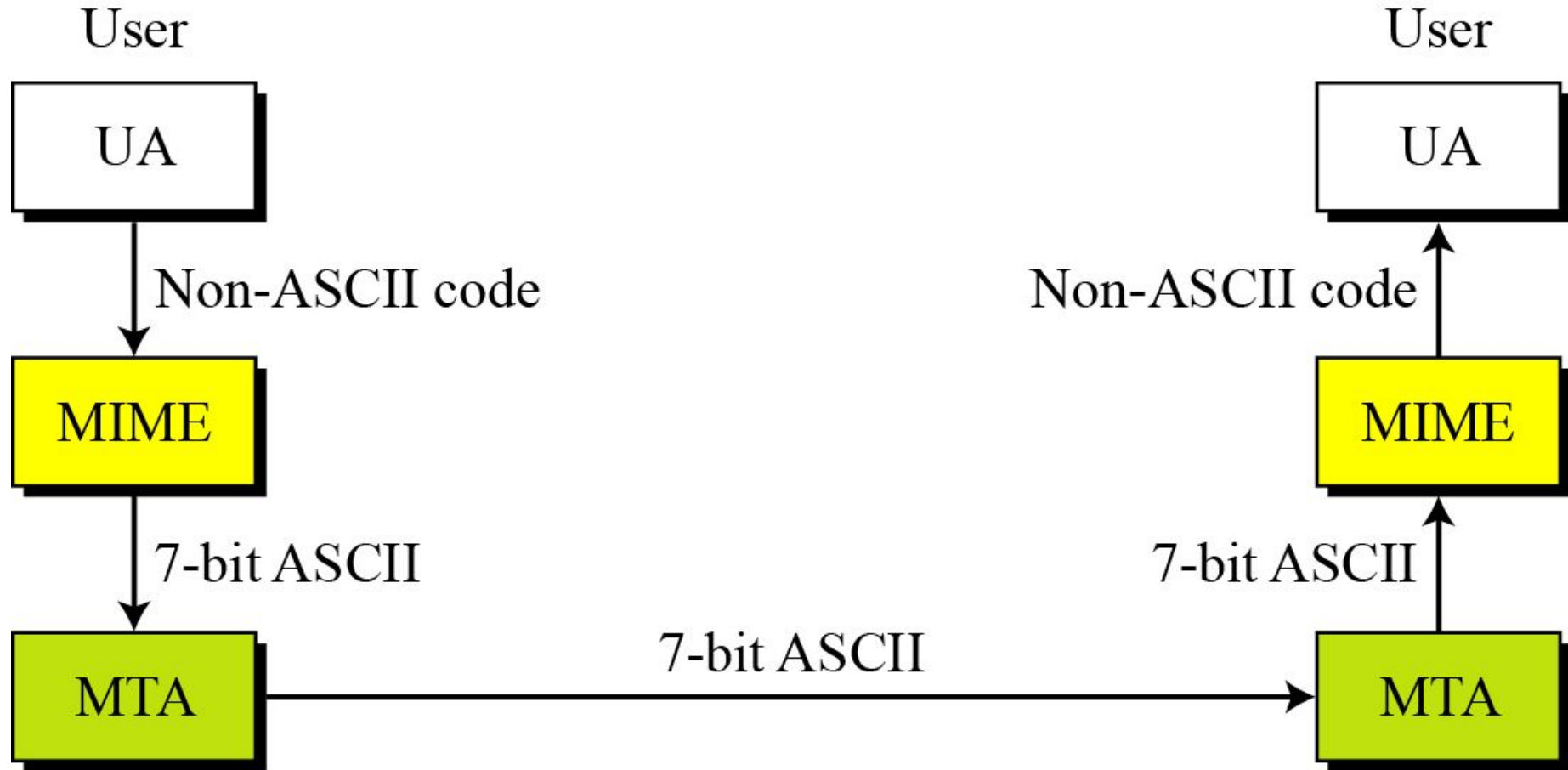
**16.3.1 MIME**

**16.3.2 S/MIME**

**16.3.3 Applications of S/MIME**

## 16.3.1 Continued

**Figure 16.23** *MIME*





## 16.3.1 *Continued*

---

E-mail header	
MIME-Version: 1.1 Content-Type: type/subtype Content-Transfer-Encoding: encoding type Content-Id: message id Content-Description: textual explanation of nontextual contents	MIME headers
E-mail body	

## 16.3.1 Continued

**Table 16.14** *Data types and subtypes in MIME*

Type	Subtype	Description
	Plain	Unformatted.
	HTML	HTML format.
Multipart	Mixed	Body contains ordered parts of different data types.
	Parallel	Same as above, but no order.
	Digest	Similar to Mixed, but the default is message/RFC822.
	Alternative	Parts are different versions of the same message.
Message	RFC822	Body is an encapsulated message.
	Partial	Body is a fragment of a bigger message.
	External-Body	Body is a reference to another message.
Image	JPEG	Image is in JPEG format.
	GIF	Image is in GIF format.
Video	MPEG	Video is in MPEG format.
Audio	Basic	Single channel encoding of voice at 8 KHz.
Application	PostScript	Adobe PostScript.
	Octet-stream	General binary data (eight-bit bytes).



## *16.3.1 Continued*

---

### **MIME-Version**

**This header defines the version of MIME used. The current version is 1.1.**

```
MIME-Version: 1.1
```

### **Content-Type**

**The content type and the content subtype are separated by a slash. Depending on the subtype, the header may contain other parameters.**

```
Content-Type: <type / subtype; parameters>
```





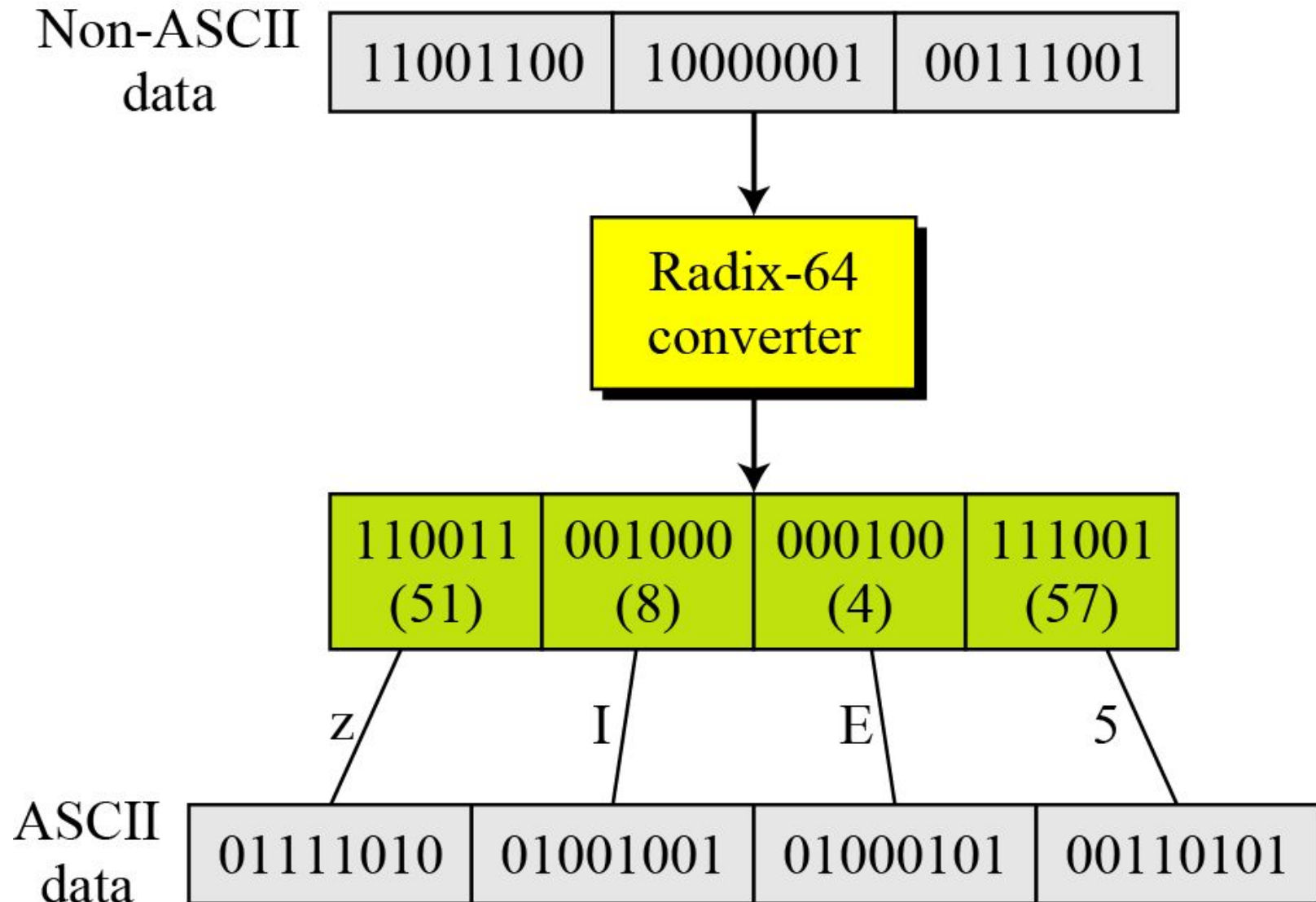
## 16.3.1 Continued

**Table 16.15** *Content-transfer-encoding*

<i>Type</i>	<i>Description</i>
7bit	NVT ASCII characters and short lines.
8bit	Non-ASCII characters and short lines.
Binary	Non-ASCII characters with unlimited-length lines.
Radix-64	6-bit blocks of data are encoded into 8-bit ASCII characters using Radix-64 conversion.
Quoted-printable	Non-ASCII characters are encoded as an equal sign followed by an ASCII code.

## 16.3.1 Continued

**Figure 16.25** *Radix-64 conversion*





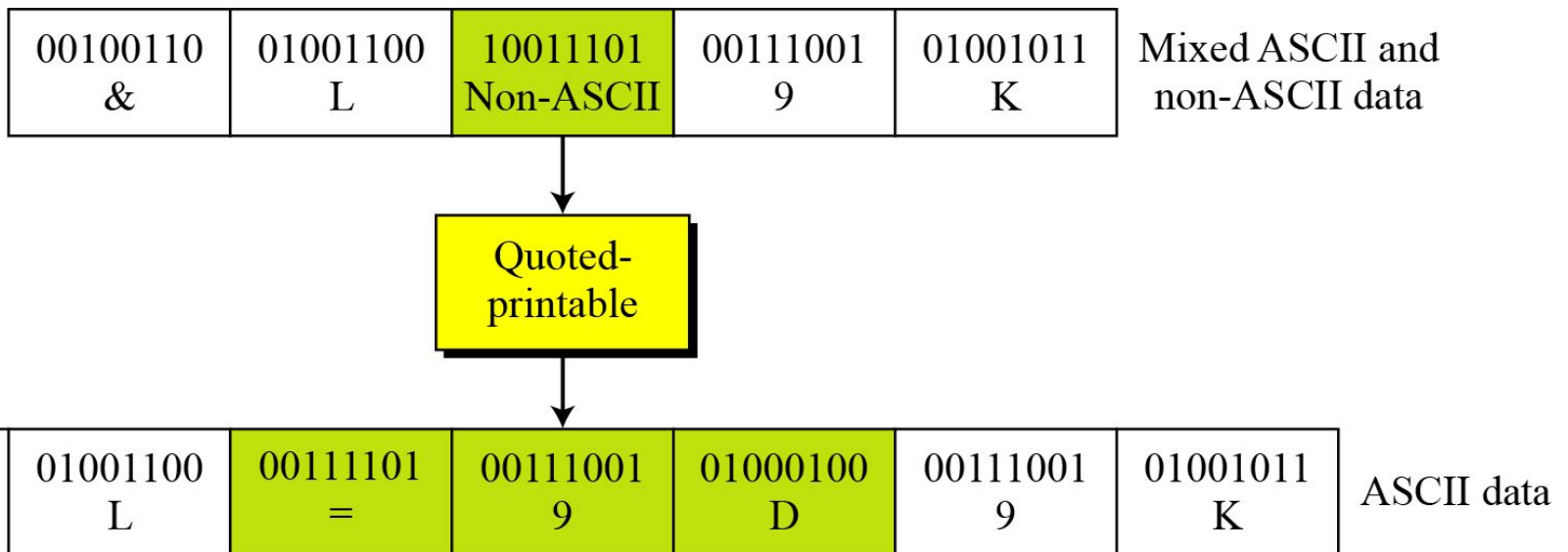
## 16.3.1 Continued

**Table 16.16** *Radix-64 encoding table*

<i>Value</i>	<i>Code</i>	<i>Value</i>	<i>Code</i>	<i>Value</i>	<i>Code</i>	<i>Value</i>	<i>Code</i>	<i>Value</i>	<i>Code</i>	<i>Value</i>	<i>Code</i>
0	<b>A</b>	11	<b>L</b>	22	<b>W</b>	33	<b>h</b>	44	<b>s</b>	55	<b>3</b>
1	<b>B</b>	12	<b>M</b>	23	<b>X</b>	34	<b>i</b>	45	<b>t</b>	56	<b>4</b>
2	<b>C</b>	13	<b>N</b>	24	<b>Y</b>	35	<b>j</b>	46	<b>u</b>	57	<b>5</b>
3	<b>D</b>	14	<b>O</b>	25	<b>Z</b>	36	<b>k</b>	47	<b>v</b>	58	<b>6</b>
4	<b>E</b>	15	<b>P</b>	26	<b>a</b>	37	<b>l</b>	48	<b>w</b>	59	<b>7</b>
5	<b>F</b>	16	<b>Q</b>	27	<b>b</b>	38	<b>m</b>	49	<b>x</b>	60	<b>8</b>
6	<b>G</b>	17	<b>R</b>	28	<b>c</b>	39	<b>n</b>	50	<b>y</b>	61	<b>9</b>
7	<b>H</b>	18	<b>S</b>	29	<b>d</b>	40	<b>o</b>	51	<b>z</b>	62	<b>+</b>
8	<b>I</b>	19	<b>T</b>	30	<b>e</b>	41	<b>p</b>	52	<b>0</b>	63	<b>/</b>
9	<b>J</b>	20	<b>U</b>	31	<b>f</b>	42	<b>q</b>	53	<b>1</b>		
10	<b>K</b>	21	<b>V</b>	32	<b>g</b>	43	<b>r</b>	54	<b>2</b>		

## 16.3.1 Continued

**Figure 16.26** *Quoted-printable*





## **16.3.2 S/MIME**

**S/MIME adds some new content types to include security services to the MIME. All of these new types include the parameter “application/pkcs7-mime,” in which “pkcs” defines “Public Key Cryptography Specification.”**

### **Cryptographic Message Syntax (CMS)**

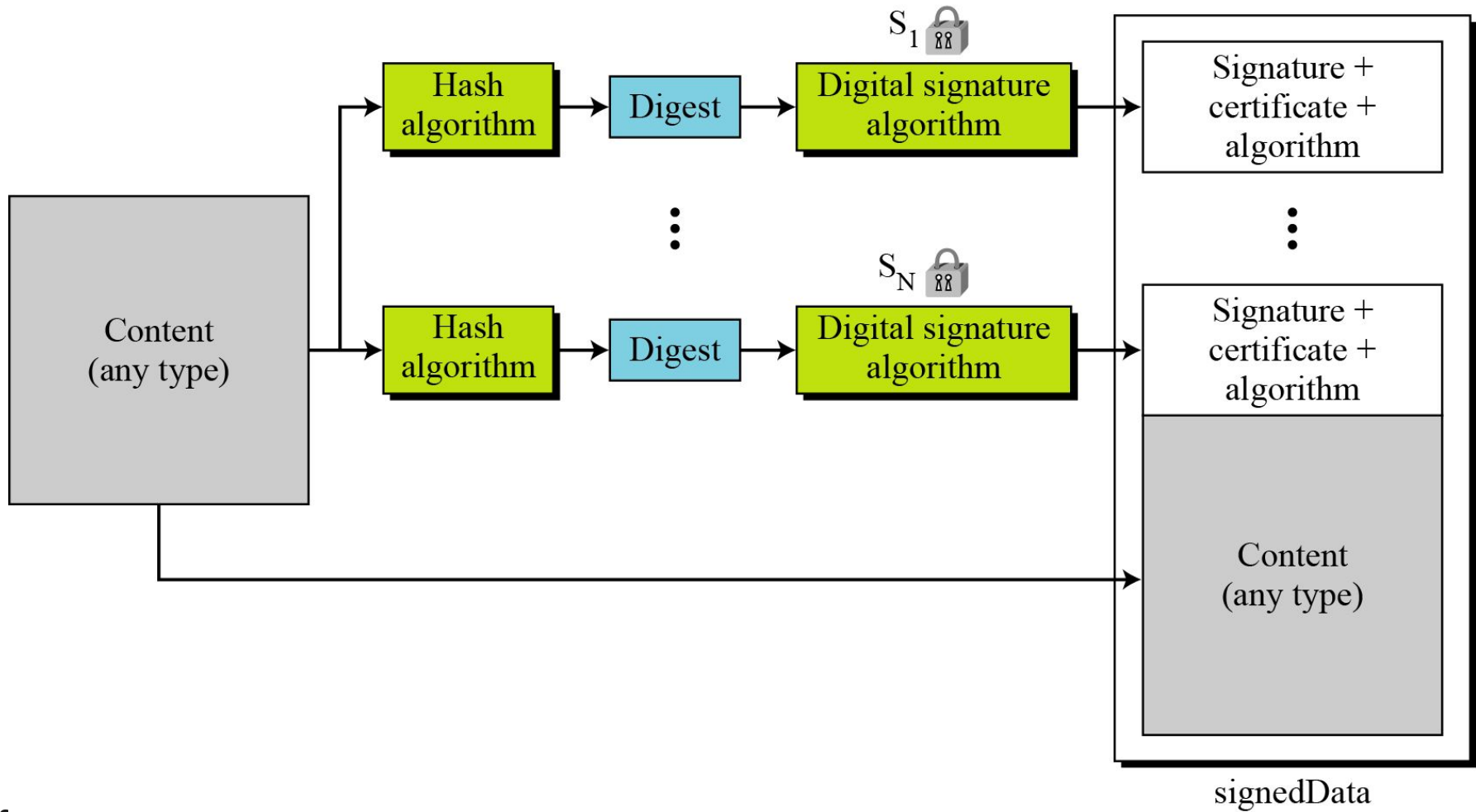
**To define how security services, such as confidentiality or integrity, can be added to MIME content types, S/MIME has defined Cryptographic Message Syntax (CMS). The syntax in each case defines the exact encoding scheme for each content type. For details, the reader is referred to RFC 3369 and 3370.**

## 16.3.2 Continued

**Figure 16.27** *Signed-data content type*

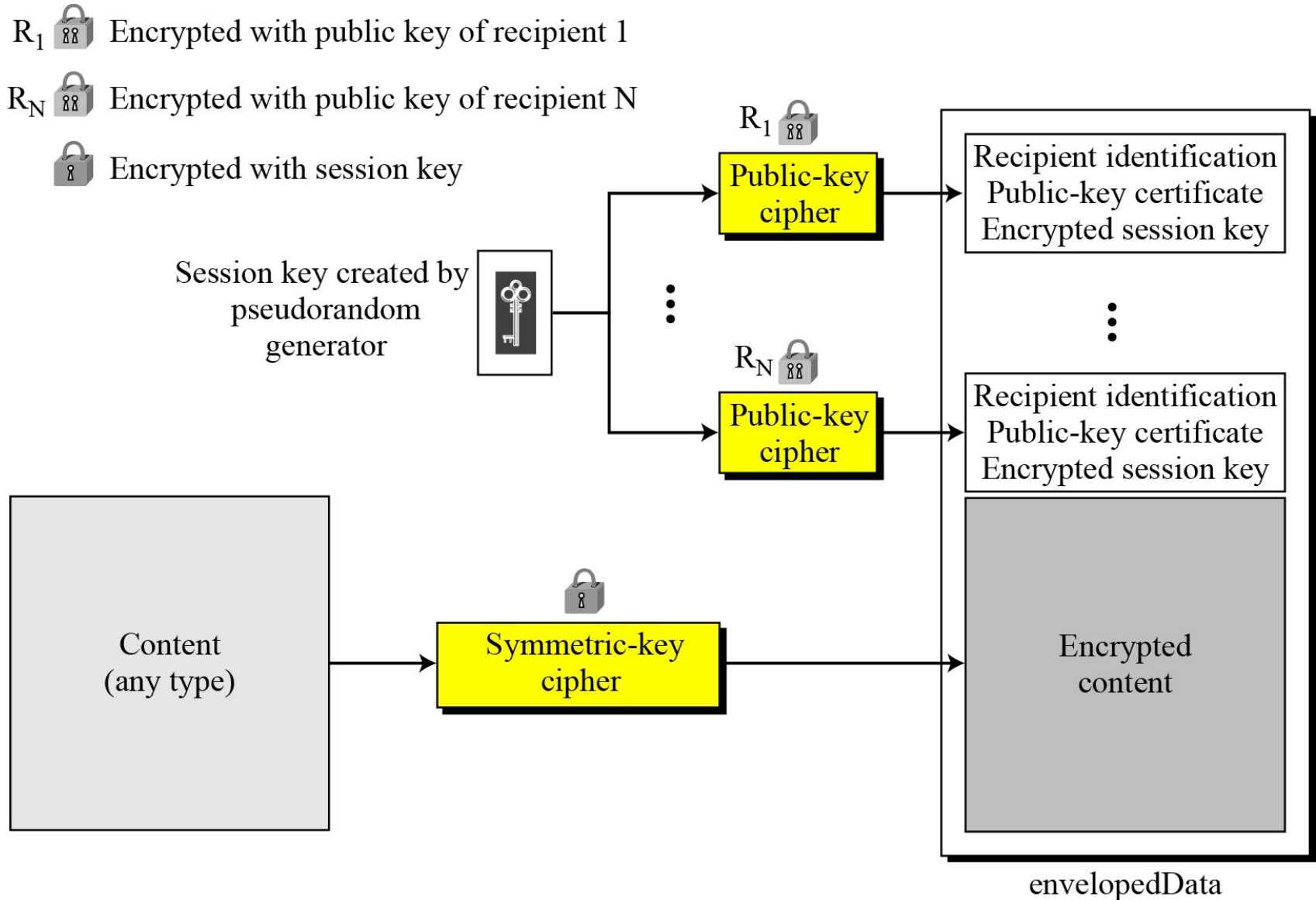
$S_1$   Signed with private key of signer 1

$S_N$   Signed with private key of signer N



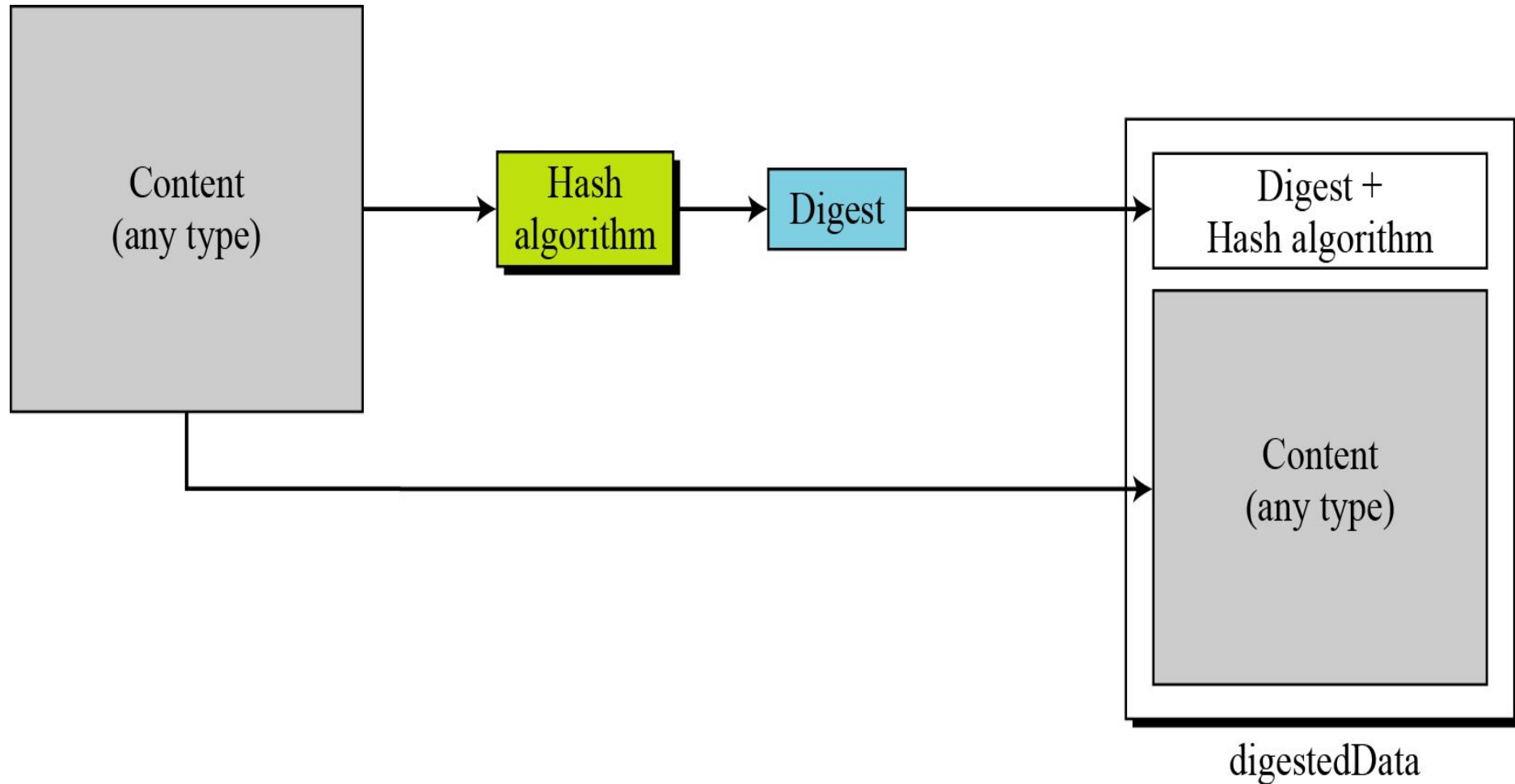
## 16.3.2 Continued

**Figure 16.28** *Enveloped-data content type*



## 16.3.2 Continued


**Figure 16.29** *Digest-data content type*




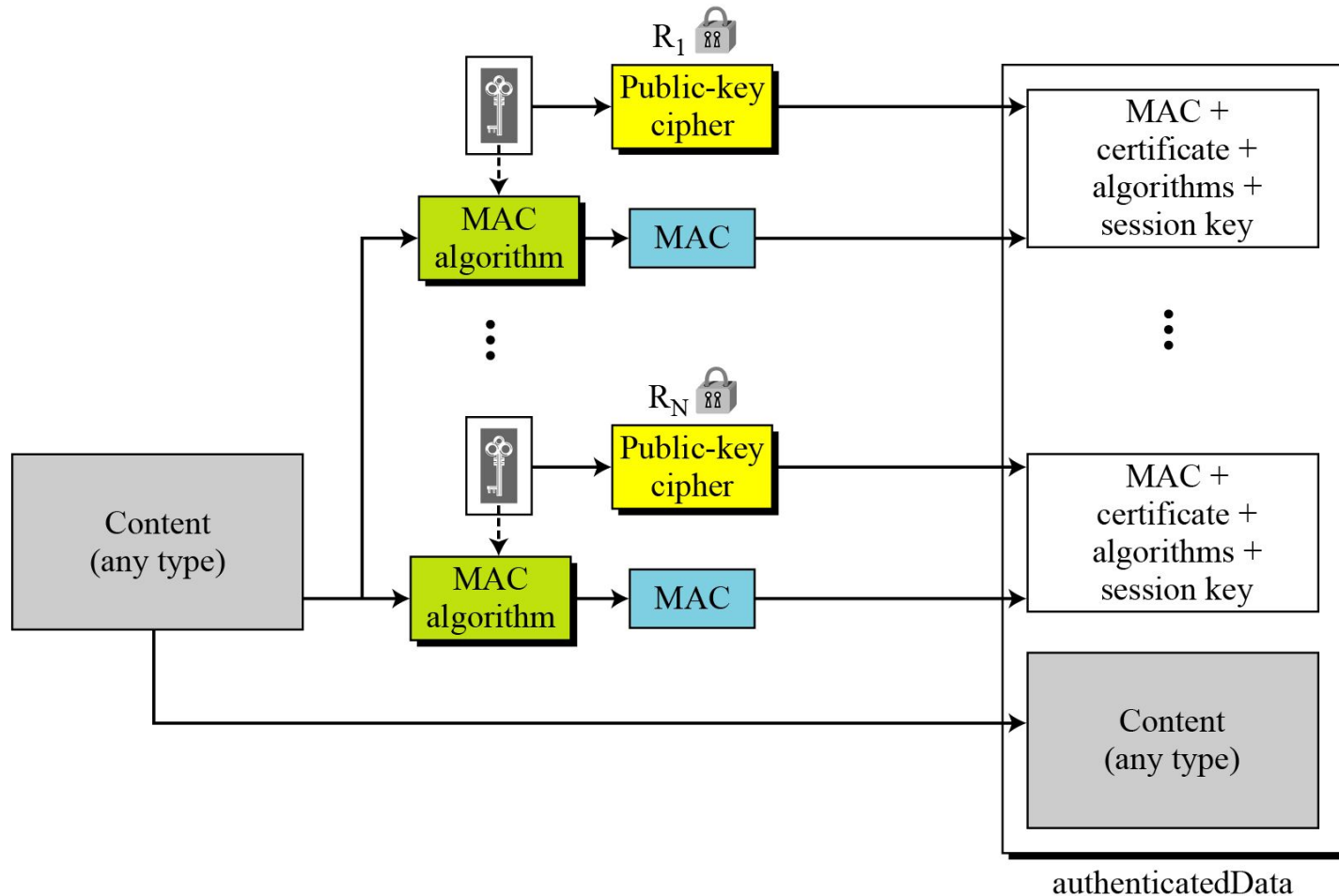


## 16.3.2 Continued

**Figure 16.30** *Authenticated-data content type*

$R_1$   Encrypted with public key of recipient 1

$R_N$   Encrypted with public key of recipient N



## 16.3.2 Continued

### Cryptographic Algorithms

S/MIME defines several cryptographic algorithms. The term “**must**” mean an absolute requirement; the term “**should**” means recommendation.

**Table 16.17** *Cryptographic algorithm for S/MIME*

<i>Algorithm</i>	<i>Sender must support</i>	<i>Receiver must support</i>	<i>Sender should support</i>	<i>Receiver should support</i>
Content-encryption algorithm	Triple DES	Triple DES		1. AES 2. RC2/40
Session-key encryption algorithm	RSA	RSA	Diffie-Hellman	Diffie-Hellman
Hash algorithm	SHA-1	SHA-1		MD5
Digest-encryption algorithm	DSS	DSS	RSA	RSA
Message-authentication algorithm		HMAC with SHA-1		